2017 IEEE 33rd International Conference on Data Engineering

Searching Time Series with Invariance to Large Amounts of Uniform Scaling

Yilin Shen*, Yanping Chen*, Eamonn Keogh†, Hongxia Jin*
*Samsung Research America, †University of California, Riverside
{yilin.shen, yanping.c, hongxia.jin}@samsung.com, eamonn@cs.ucr.edu

***Abstract*—Similarity search is arguably the most important primitive in time series data mining. Recent research has made significant progress on fast algorithms for time series similarity search under Dynamic Time Warping (DTW) and Uniform Scaling (US) distance measures. However, the current state-of-the-art algorithms cannot support greater amounts of rescaling in many practical applications. In this paper, we introduce a novel lower bound, $\mathsf{LB}_{\mathsf{new}}$, to allow efficient search even in domains that exhibit more than a factor-of-two variability in scale. The effectiveness of our idea is validated on various large-scale real datasets from commercial important domains.**

***Index Terms*—Lower Bounds, Time Series Analytics, Dynamic Time Warping, Uniform Scaling**

I. Introduction

Time series similarity search is useful in its own right as an exploratory tool, and it is used the core subroutine in almost all higher-level algorithms, such as rule discovery, motif discovery, anomaly detection, classification, clustering and summarization. Given this and the ubiquity of time series data, the last decade has seen significant progress in research efforts to speed up similarity search, and we now have very fast algorithms for time series similarity search under both the Dynamic Time Warping (DTW) [15], and Uniform Scaling (US) distance measures [11].

The current state-of-the-art algorithm for US has only been demonstrated for the modest amounts of rescaling encountered in datasets produced by human behaviors, in which we will only face small amounts of scaling, i.e., 10% to 15%. However, in many industrial and commercial contexts, such as the music retrieval [5] and the biometrics [11], [16], we may encounter much greater amounts of rescaling, rendering current solutions little better than brute force search.

As such, the requirement of both DTW and US invariance significantly increases computational complexity due to the calculation of DTW distance between the query pattern and many different sizes of each prefix. On the other hand, unfortunately, the current state-of-the-art lower bounds perform little better than brute force search in this case [7], [10] (later confirmed in experiment). Therefore, it is critically desirable to design an efficient algorithm to speed up the searching of large-scale time series with invariance to large amounts of uniform scaling.

In this paper, in order to mitigate this problem, we introduce a novel lower bound $\mathsf{LB}_{\mathsf{new}}$, which *for the first time* allows efficient search even in domains that exhibit more than a factor-of-two variability in scale. The contributions and advantages of our method can be summarized as follows:

- We design $\mathsf{LB}_{\mathsf{new}}$ based on envelope-based lower-bounding. This means that our method can be "plugged in" to any of the dozens of envelope-based indexing approaches (see [5] and the references therein);
- We develop different variations of $\mathsf{LB}_{\mathsf{new}}$ to handle the combination of DTW and US (USDTW), along with an efficient calculation algorithm and theoretical guarantees.
- Extensive experiments on very large real world datasets show that the application of $\mathsf{LB}_{\mathsf{new}}$ can improve the searching of time series with more than a factor-of-two variability in scale up to *27 times faster* than using the state-of-the-art lower bound.

The rest of paper is organized as follows: Related work is summarized in Section II. Section III discusses our novel lower bound $\mathsf{LB}_{\mathsf{new}}$ on USDTW metric and its extension to US metric. The experimental results are presented in Section IV and Section V concludes the whole paper.

II. Related Work

Time series subsequence searching has become a key driver for a majority of time series mining approaches (e.g., classification, clustering, etc.). Due to the space limit, the interested readers are suggested to read two comprehensive reviews of existing work [6], [14]. In order to speed up time series similarity search, many researchers have focused on proposing lower bounds to admissibly prune off unpromising candidates. Among all these exact lower bounds, $\mathsf{LB}_{\mathsf{Keogh}}$ [10] has been shown to be most efficient in the literature. Recently, Rakthanmanon *et al.* [15] proposed UCR-DTW to combine all acceleration approaches together for indexing time series using DTW distance. However, these approaches are only designed to lower bounding DTW distance, thus not applicable under USDTW distance. The most relevant work was proposed by Fu *et al.* [7], in which $\mathsf{LB}_{\mathsf{Keogh}}$ is extended to support US and USDTW. Unfortunately, all these approaches only provide very loose lower bounds for time series with invariance to large amounts of uniform scaling, which therefore fail to accelerate the searching in many practical cases.

III. Lower Bound $\mathsf{LB}_{\mathsf{NEW}}$ & Algorithm

In this section, we propose a novel lower bound $\mathsf{LB}_{\mathsf{new}}$ with theoretical correctness and performance proofs. Besides, we introduce an efficient $\mathsf{LB}_{\mathsf{new}}$ calculation algorithm and analyze

2375-026X/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDE.2017.52

its time and space complexity. Specifically, we discuss the lower bound LB-UD$_{\text{new}}$ on (U)S(D)TW distance first and then extend it into LB-U$_{\text{new}}$ on (U)S distance metric.

To begin with, we first briefly introduce the USDTW distance and drawbacks of existing approaches. USDTW considers all prefix subsequences of C of length from $\lceil m/l \rceil$ to $\min(\lceil lm \rceil, n)$, where $|Q| = m$, $|C| = n$ and l is a scaling factor bound ($l \geq 1$). And USDTW calculates DTW distances between each rescaled prefix of C to the query sequence Q and outputs the minimum DTW distance. However, existing LB-UD$_{\text{Keogh}}$ [7] on USDTW (Extension of LB$_{\text{Keogh}}$) suffers from two drawbacks: (1) it requires the on-the-fly maintenance of envelop on C that is most infeasible due to high computational costs; (2) the lower bound is too loose to prune enough subsequence in C, which makes the searching only little bit better than brute force search.

A. Main Idea & Mathematical Formulation

1) Idea & Intuition: We consider constructing an envelop on the query sequence Q instead of on the testing sequence C, such that this only needs preprocessing and dramatically reduces the running time. The main idea is two-fold: (1) we consider using the largest alignment factor $L = \min(\lceil lm \rceil, n)$; (2) we consider the lower bound between each prefix of C and Q without the scaling up of C. In this case, we do not have to calculate the lower bound of each DTW from scratch and therefore dramatically reduce the computational cost.

2) Mathematical Formulation of LB-UD$_{new}$: Consider the testing sequence C of length $|C| = n$ and scaling factor $l \geq 1$. For the query sequence Q of length $|Q| = m$, we only need to consider the prefix $C(k)$ of C where $\lceil m/l \rceil \leq k \leq \min(\lceil lm \rceil, n)$. Thus, we define a corresponding indexed collection $\mathbb{Q} = (\mathbb{q}_1, \ldots, \mathbb{q}_{\min(\lceil lm \rceil, n)})$ of sequences in which each element is as follows:

$$\mathbb{q}_j = (q_{\lceil j/l \rceil - r}, \ldots, q_{\lceil jl \rceil + r}) \tag{1}$$

Considering the boundary cases, $\mathbb{q}_j$ can be rewritten as:

$$\mathbb{q}_j = (q_{\max(1, \lceil j/l \rceil - r)}, \ldots, q_{\min(\lceil jl \rceil + r, m)}) \tag{2}$$

This definition of $\mathbb{Q}$ takes into account both scaling factor $l \geq 1$ and DTW constraint parameter $r \geq 0$. Here r is a DTW constraint parameter which defines the reach, or allowed range of warping.

To this end, we can mathematically present our lower bound LB-UD$_{\text{new}}$ as follows:

$$\text{LB-UD}_{\text{new}} = \sum_{j=1}^{\min(\lceil lm \rceil, n)} \delta(c_j, \mathbb{q}_j) \tag{3}$$

where $\delta(x, Y)$ is the smallest distance among all the distances between each element $y \in Y$ to x for an element x and a sequence Y, defined as $\delta(x, Y) = \min_{y \in Y} d(x, y)$. Specifically, for each element c_j in the largest scalable prefix $C(1 : \min(\lceil lm \rceil, n))$ of C, we find the element in $\mathbb{q}_j$ with smallest distance to c_j and sum up all distances.

Algorithm 1: $\delta(d, \tilde{X})$ Calculation

Input : data point d, sorted data sequence $\tilde{X}$
Output: $\delta(d, \tilde{X})$

```
// check the necessity of binary search
if d ≤ X̃(1) then
    δ(d, X̃) ← |d − X̃(1)|²;
else if d ≥ X̃(|X̃|) then
    δ(d, X̃) ← |d − X̃(|X̃|)|²;
// do binary search if needed
else
    δ(d, X̃) ← min_{s∈X̃} |d − X̃(s)|² using binary search on x and X̃;
return δ(d, X̃);
```

Algorithm 2: LB-UD$_{\text{new}}(Q, C, l, r)$ Calculation

Input : query sequence Q, data sequence C, scaling factor l, DTW constraint parameter r
Output: LB-UD$_{\text{new}}(Q, C, r)$

```
1  // preprocessing: construct indexed collection ℚ w.r.t. query sequence Q
2  for j ← 1 to min(⌈lm⌉, n) do
3      q̃_j ← sorted sequence ℚ_j;
4  // calculate lower bound for shortest prefix of C
5  for j ← 1 to ⌈m/l⌉ do
6      Calculate δ(c_j, q̃_j) using Algorithm 1;
7      δ(c_j, ℚ_j) ← δ(c_j, q̃_j);
8  LB-UD_new(Q, C, l, r) ← Σ_{j=1}^{min(⌈m/l⌉, n)} δ(c_j, ℚ_j);
9  // calculate each lower bound incrementally
10 for j ← ⌈m/l⌉ to min(⌈lm⌉, n) do
11     Calculate δ(c_j, q̃_j) using Algorithm 1;
12     δ(c_j, ℚ_j) ← δ(c_j, q̃_j);
13     LB-UD_new(Q, C, l, r) ← LB-UD_new(Q, C, l, r) + δ(c_j, ℚ_j);
14 return LB-UD_new(Q, C, l, r);
```

B. A Novel Incremental LB-UD$_{new}$ Calculation Algorithm

We first take a closer look at the intuition behind LB-UD$_{\text{new}}$. In fact, $\sum_{j=1}^{k} \delta(c_j, \mathbb{q}_j)$ is actually a slightly looser lower bound of DTW between aligned Q and prefix $C(1:k)$ of C (We will formally prove this in Section III-C). This motivates the following lightweight incremental LB-UD$_{\text{new}}$ calculation algorithm.

As shown in Algorithm 2, we first do preprocessing to obtain the sorted $\tilde{\mathbb{q}}_j$. Then, in line 3-6, we initialize LB-UD$_{\text{new}}$ by calculating a looser lower bound of DTW between the shortest prefix $C(1 : \lceil m/l \rceil)$ of C and query sequence Q, by using $\mathbb{Q}$ with a scaling factor $l > 1$. Next, for each c_j with $\lceil m/l \rceil + 1 \leq j \leq \min(\lceil lm \rceil, n)$, we increment LB-UD$_{\text{new}}$ by adding on each $\delta(c_j, \mathbb{q}_j)$.

Note that our algorithm is designed to support breaking out of the loop in line 7-10 at any time, as long as the lower bound is large enough. For instance, if LB-UD$_{\text{new}}$ is used to prune off unpromising candidates during the searching of nearest neighbor subsequence, we can break this loop once the lower bound is larger than best-so-far value.

Time Complexity: As line 1-2 in Algorithm 2 is pre-processed, we again focus on analyzing the time complexity in line 3-6. In iteration j, since the length of $\mathbb{q}_j$ is $(l-1/l)j+2r$, the time complexity of line 5 and line 8 are $\log[(l-1/l)j+2r]$ if binary search is needed and $O(1)$ otherwise. Thus, the total time complexity from line 3-5 is $O(lm\log(lm+2r))$.

Space Complexity: The only space needed is to save all sorted sequence $\tilde{\mathbb{q}}_j$ of the indexed collection $\mathbb{Q}$. Since each $\tilde{\mathbb{q}}_j$ can be at most of size $(l-1/l)j+2r$, the space complexity is $O(l^3m+rlm)$.

C. Theoretical Guarantees

In order to show the correctness of LB-UD$_{\text{new}}$, we need to prove that $\sum_{j=1}^{k}\delta(c_j,\mathbb{q}_j)$ is a lower bound of $\text{DTW}_r(C^{\min(\lceil lm\rceil,n)}(1:k),Q^{\min(\lceil lm\rceil,n)})$ for any $\lceil m/l\rceil+1\le k\le\min(\lceil lm\rceil,n)$.

Theorem 1. *For any $\lceil m/l\rceil+1\le k\le\min(\lceil lm\rceil,n)$, $DTW_r(C^{\min(\lceil lm\rceil,n)}(1:k),Q^{\min(\lceil lm\rceil,n)})$ is always lower bounded by $\sum_{j=1}^{k}\delta(c_j,\mathbb{q}_j)$.*

Proof. Let k be an arbitrary integer between $\lceil m/l\rceil+1$ and $\min(\lceil lm\rceil,n)$. We denote the rescaled sequences $C^{\min(\lceil lm\rceil,n)}(1:k)=(c'_1,\ldots,c'_{\min(\lceil lm\rceil,n)})$ and $\mathbb{q}'_j$ as DTW envelop w.r.t. $Q^{\min(\lceil lm\rceil,n)}$. We prove as follows:

$$\text{DTW}_r(C^{\min(\lceil lm\rceil,n)}(1:k),Q^{\min(\lceil lm\rceil,n)})$$
$$\ge\sum_{j=1}^{\min(\lceil lm\rceil,n)}\delta(c'_j,\mathbb{q}'_j)\ge\sum_{j=1}^{k}\delta(c_j,\mathbb{q}_j)$$

Particularly, the second step holds since $C^{\min(\lceil lm\rceil,n)}(1:k)$ only contains duplicated elements in $C(1:k)$ by doing nearest neighbor interpolation and the elements in $\mathbb{q}_j$ is a superset of elements in $\mathbb{q}'_j$ for each j. □

D. Tightness of LB-UD$_{new}$

The perceptive reader may have noticed that by considering USDTW distance, our proposed lower bound LB-UD$_{\text{new}}$ rank objects without compensating for potentially different lengths. One may observe that this might not be practically useful since it contracts the conventional wisdom of say, the string processing community. For example, while HammingDist(to, so) and HammingDist(ridiculous, nidiculous) are both one, the latter seems to be more similar. To compensate, the Hamming distance is typically adjusted by dividing by the length of the strings. However, we cannot directly normalize for length in the case of time series [12]. In general, the Euclidean distance does not "grow" linearly with the length of the sequences being considered. Fortunately, the good news is that it does not seem to matter for most datasets. In general, the change in similarity when the right length is considered simply dwarfs all other considerations. This has been independently noted in [12], and confirmed in our experiments by showing the efficiency of our novel lower bound LB-UD$_{\text{new}}$.

E. Discussion: Applying onto Uniform Scaling

It is easy to observe that our novel LB-UD$_{\text{new}}$ can be simply extended to LB-U$_{\text{new}}$ for US distance by defining the elements in $\mathbb{Q}^{\text{US}}$ slightly differently as follows:

$$\mathbb{q}_j^{\text{US}}=(q_{\max(1,\lceil j/l\rceil)},\ldots,q_{\min(\lceil jl\rceil,m)}) \tag{4}$$

With this tweaked envelop, both Algorithm 2 and theoretical analysis can be applied directly without additional modification.

IV. Experimental Evaluation

In this section, our focus is to show the efficiency of novel lower bounds LB$_{\text{new}}$, since our lower bounds are exact lower bounds with theoretical proofs.

We consider two large datasets: the dishwasher electrical load dataset (Section IV-A) and the BIDMC Congestive Heart Failure Database [1] (Section IV-B).

We evaluate LB-UD$_{\text{new}}$ against the existing LB-UD$_{\text{Keogh}}$ [7] on USDTW (Extension of LB$_{\text{Keogh}}$). Note that in the experiment we focus on the comparison with current exact searching approaches that are guaranteed no false negative.

In the experiment, the query Q is created by the average of all positive subsequences scaled to the same length. The uniform scaling factor l and DTW constraint parameter r are then selected as the largest factors between true subsequence and created query pattern. We test the experiments on personal computer which is equipped with 2.6GHz CPU and 8GB RAM. Due to the long running time, we run each experiment 3 times and report the average result.

A. Case Study 1: Electrical Load Mining

We consider a case study in exploratory querying residential electrical loads. Similarity search in this domain is often a frequently invoked subroutine to support energy desegregation [8]. We begin by examining the first few days of House-1 from the REFIT dataset [13]. This dataset contains 7,633,070 datapoints, corresponding to two years of data sampled every 8 seconds.

In the dataset, we found multiple instances of a similar pattern in the first few days of this dataset, and used them to form an idealized prototype to query the rest of the data. We observed that there is some variability in the patterns. This could be addressed with DTW, although for such well-defined patterns, Euclidean distance can suffice [6]. If we use our prototype query to do top-twenty similarity-search on the remainder of the data, the results using either Euclidean distance or DTW are similar.

On a hunch, we searched the data again, this time with an extremely large amount of scaling, up to 250%. To our surprise, as Fig. 1 shows, the dataset *does* contain several such drastically rescaled examples of the pattern in question.

All of the above is simply to justify the need for significant amounts of scaling in the domain. The result shows that the application of LB-UD$_{\text{new}}$ takes only 2.5 hours, which is 1.25 times faster than using LB-UD$_{\text{Keogh}}$.

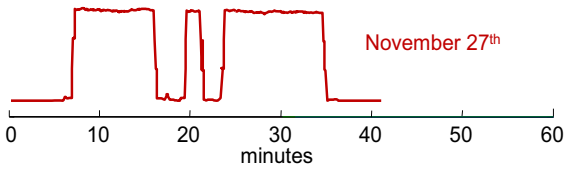

Fig. 1: An event discovered is a near perfect match to the ideal query Q if Q is stretched by 204%.

B. Case Study 2: Electrocardiogram Mining

We consider another case study in querying ECG data streams for monitoring patients with severe congestive heart failure, which is also enabled by similarity search. Due to the large scale dataset, we begin with monitoring one patient out of 15 patients in the BIDMC Congestive Heart Failure Database [1]. The data contains 17,998,848 datapoints, recording about 20 hours in duration sampled at 250 Hz.

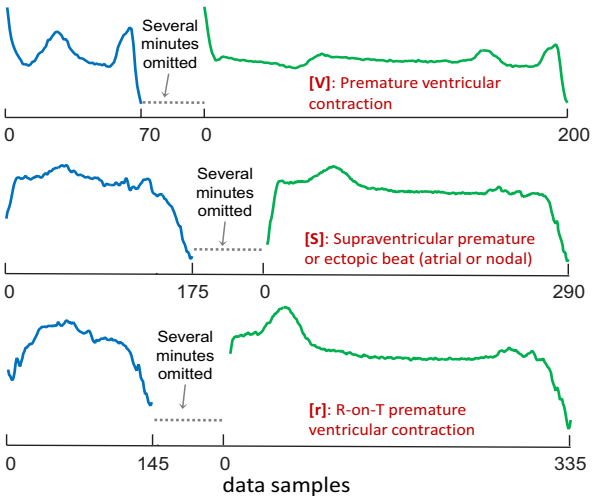

Fig. 2: Two samples of heartbeats in each type of beat annotations: top) Premature ventricular contraction (V) with rescaling factor 285%; middle) Supraventricular premature or ectopic beat (atrial or nodal) (S) with rescaling factor 165%; bottom) R-on-T premature ventricular contraction (r) with rescaling factor 231%.

Similarly, Fig. 2 shows an extremely large amount of scaling (up to 285%) of heartbeats in a patient's ECG data. In order to monitor the heart failures with different annotations, we use the first 10 mins data samples to form each idealized prototype of each annotation (V, S, and r) to query the rest of data. Due to the space limitation, we do not show the generated idealized query and discovered event again as they are similar as those in Figure 2.

The result shows that the application of LB-UD$_{\text{new}}$ consistently takes least running time, that speeds up the state-of-the-art algorithm (using LB-UD$_{\text{Keogh}}$) up to *27 times*.

V. Conclusion

We proposed novel lower bounds LB$_{\text{new}}$ to accelerate the searching of time series with large amounts of rescaling in practice. LB$_{\text{new}}$ is proposed to support both USDTW and US distance metrics, along with efficient calculation algorithms and theoretical guarantees. We demonstrated on various very large real-world datasets that the application of our lower bounds can speed up searching time series up to 27 times, rendering it very useful in practical monitoring applications. Thanks to the envelop property of all variations of LB$_{\text{new}}$, they can be simply "plugged in" existing high-level time series mining algorithms in their searching primitive subroutine, with significant reduction of running time.

References

[1] The bidmc congestive heart failure database. http://www.physionet.org/physiobank/database/chfdb/.
[2] Pamap, physical activity monitoring for aging people. http://www.pamap.org/demo.html, retrieved 2016-04-04.
[3] N. Begum, L. Ulanova, J. Wang, and E. Keogh. Accelerating dynamic time warping clustering with a novel admissible pruning strategy. KDD '15, pages 49–58, New York, NY, USA, 2015. ACM.
[4] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista. The ucr time series classification archive, July 2015. www.cs.ucr.edu/~eamonn/time_series_data/.
[5] R. B. Dannenberg, W. P. Birmingham, G. P. Tzanetakis, C. P. Meek, N. P. Hu, and B. P. Pardo. The musart testbed for query-by-humming evaluation. *Comput. Music J.*, 28(2):34–48, June 2004.
[6] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and mining of time series data: Experimental comparison of representations and distance measures. *Proc. VLDB Endow.*, 1(2):1542–1552, Aug. 2008.
[7] A. W.-C. Fu, E. Keogh, L. Y. Lau, C. A. Ratanamahatana, and R. C.-W. Wong. Scaling and time warping in time series querying. *The VLDB Journal*, 17(4):899–921, July 2008.
[8] S. Gupta, M. S. Reynolds, and S. N. Patel. Electrisense: Single-point sensing using emi for electrical event detection and classification in the home. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, UbiComp '10, pages 139–148, New York, NY, USA, 2010. ACM.
[9] B. Hu, Y. Chen, and E. J. Keogh. Time series classification under more realistic assumptions. In *Proceedings of the 13th SIAM International Conference on Data Mining, May 2-4, 2013. Austin, Texas, USA.*, pages 578–586, 2013.
[10] E. Keogh. Exact indexing of dynamic time warping. In *Proceedings of the 28th International Conference on Very Large Data Bases*, VLDB '02, pages 406–417. VLDB Endowment, 2002.
[11] E. Keogh, T. Palpanas, V. B. Zordan, D. Gunopulos, and M. Cardle. Indexing large human-motion databases. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB '04, pages 780–791. VLDB Endowment, 2004.
[12] A. Mueen. Enumeration of time series motifs of all lengths. In *2013 IEEE 13th International Conference on Data Mining*, pages 547–556, Dec 2013.
[13] D. Murray, J. Liao, L. Stankovic, V. Stankovic, R. Hauxwell-Baldwin, C. Wilson, M. Coleman, T. Kane, and S. Firth. *A data management platform for personalised real-time energy feedback*. 8 2015.
[14] P. Papapetrou, V. Athitsos, M. Potamias, G. Kollios, and D. Gunopulos. Embedding-based subsequence matching in time-series databases. *ACM Trans. Database Syst.*, 36(3):17:1–17:39, Aug. 2011.
[15] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. KDD '12, pages 262–270, New York, NY, USA, 2012. ACM.
[16] R. Tanawongsuwan and A. Bobick. Modelling the effects of walking speed on appearance-based gait recognition. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–783–II–790 Vol.2, June 2004.